GETTING STARTED WITH HSI AT NCEP

George VandenBerghe
IBM
Nov. 25, 2003

TOPICS

- HSI command and performance
- Interactive commands
- Comparison with ftp
- Batch commands
- File Pipes
- Examples (including file pipes with tar)
- Filters and text pipes
- Cautions and caveats
- Summary.

INTRODUCTION

- HSI IS A MORE UNIX LIKE INTERFACE TO HPSS
- HAS CHARACTERISTICS OF REMOTE SHELLS IN BATCH JOBS AND INTERACTIVE SHELLS OR SCRIPTS IN INTERACTIVE JOBS
- NOT 1:1 MAPPING BETWEEN UNIX AND HSI BUT MUCH CLOSER FIT THAN FTP
- MUCH MORE POWERFUL RECURSION

DOCUMENTATION

- http://ibmdocs.ncep.noaa.gov/userman/SDSC.HSI.guide.htm
- this is full documentation from HPSS support
- http://ibmdocs.ncep.noaa.gov/userman/hsi.commands.html
- This is a summary of commands from HPSS support
- http://ibmdocs.ncep.noaa.gov/userman under **HSI** (this is a pdf)

WHERE HSI HELPS

- HSI is most useful for file and directory manipulation.
- HSI wildcards are superior to other interfaces'
- No advantages to large file transfers.
- Htar is generally better for working with tarballs.

PERFORMANCE

- Large file performance is comparable to pftp or well configured htar. Don't worry about it.
- Small file performance is bound by HPSS nameserver speed (as is ftp). This is a few tens of names per second (slower than GPFS or NFS).
- Htar is not part of HSI. When working with tarballs hsi is good for managing them as data objects while htar is better for managing their contents.
- Because of seek capability in tarballs htar does less work and is therefore faster.

A WORD ABOUT PERFORMANCE

- The htar utility can access a tarball at random. Unix tar must stream through the entire tarball. If extracting a small fraction of the tarball from a tarball on HPSS disk, htar is MUCH MUCH faster.
- For a tarball on tape, htar seek is useless. However then, htar will stop when the desired file is extracted from the tarball. Average read savings is 50%. Tar would go through the WHOLE tarball.
- Htar 10-100x faster for small extracts from disk
- Htar averages 2x faster for single small extracts from tape. For multiple small extracts scattered thru tarball htar approaches unix tar speed.
- Htar can be significantly slower if its .idx file gets migrated. (two tape mounts rather than one)

INTERACTIVE USE

- HSI is on f1n1 and s1n1 only.. Will soon be on other nodes.
- It is in /usr/bin which is in default path. Users only need to enter

  hsi
- To get placed in the "hsi shell interface"
- hsi –help presents a very good nutshell description of the commands.

INTERACTIVE USE

- Started with “hsi” command on f1n1/s1n1
- *(on other nodes this is /usrx/local/hpss/hsi/bin/hsi)*
- This puts user in an hsi “shell”
- HSI commands ls, du, mkdir, find, rm, cp, etc are analogous to unix (a few options don’t work e.g. find -exec {})
- Some comands have no unix analogs (stage)
- HSI ←---→ frost/snow transfers look similar to ftp but without login.
- Transfer commands are get and put
- Syntax is different from ftp.. HSI file is always on RHS of transfer pair

FTP V.S. HSI TRANSFER

- FTP SYNTAX

put localfile remote

get remote localfile

- HSI SYNTAX

put localfile : hsifile

get localfile : hsifile

- The COLON is significant and localfile is always on LHS of the argument pair.

DIRECTORY NAVIGATION

- HSI is like ftp and pftp
- You are placed in your HPSS home directory at HSI startup.
- ***(at least one user has reported this is erroneous)***
- You do not need the mandatory full path names as you did with htar.
- As in unix and ftp you can use full and relative path names.

HSI COMMANDS

- Many support recursive options of unix e.g. ls –R which is unavailable in pftp
- Find is available. This is enormous improvement over ftp.
- Pipes can replace local or remote files in HSI commands. This allows cat into and out of HSI This is useful for the tar pipes familiar to HSM (and Cray) users.

HSI COMMANDS

- HSI SHELL HAS "l" prefix options. These act on host rather than HPSS
- **cd** changes HPSS directory
- **lcd** changes LOCAL directory (ftp does this too)
- **lls** lists a LOCAL file while ls lists the HPSS file.

HSI BATCH

- HSI can be run in scripts.
- Individual commands can be issued as "one liners"
- Syntax is **hsi " command ; command2 .."**
- Example **hsi "cd fcst.dir ; ls –l"**
- These are succinct for few commands. *Costly for large numbers of commands.*
- Each one liner is a NEW session and requires ~ 1 sec to authenticate and clean up

HSI BATCH

- HSI can take commands from here files

```
hsi <<EOF
pwd
cd dir
ls –l
get /gpfstmp/frostfile : hpssfile
EOF
```

HSI BATCH

- HSI CAN ALSO RUN ITS OWN SCRIPTS OR MACROS.
- SYNTAX FOR THESE IS
- **hsi "in cmdfile"** where cmdfile is a file containing hsi commands

EXAMPLES

- File transfer
- put /gpfstmp/stuff : /hpssuser/g02/wx15gv/junkstuff
- get /gpfstmp/stuff : /hpssuser/g02/wx15gv/junkstuff
- Note the local file is on the LHS and HPSS file is on RHS and
- NOTE THE COLONS

EXAMPLES

- **Ls –Rl**

  ! THIS LISTS ALL FILES INCLUDING SUBDIRECTORY FILES IN YOUR DIRECTORY.

hsi "find . -print"

! THIS LISTS ALL OF YOUR FILES AND DIRECTORIES IN A TREE FORM

hsi " find . -type d -print"

! THIS LISTS ALL OF YOUR DIRECTORIES

hsi " find . -type f -size +2000000 -print"

! THIS FINDS FILES LARGER THAN 1GB

PIPE EXAMPLES

- **cd /gpfstmp/dir/yourstuff ; tar -cvf - . | hsi " put - : youtstuff.tar"**

THIS WRITES A TAR ARCHIVE TO STDOUT AND HSI INGESTS IT INTO HPSS FILE yourstuff.tar

- **hsi "get - : yourstuff.tar" |tar -tvf –**

THIS DOES INVENTORY OF HPSS FILE yourstuff.tar

- **hsi "get - : yourstuff.tar" |tar -xvf - a.file**

This extracts file a.file from the tarball.

Note replacement of local file name with -
for stdin and stdout in the three pipes.

INTERACTIVE PIPE EXAMPLES

- Syntax for these is tricky and nonintuitive but simple once understood.
- Interactive put is
- **put "| localshell's output" : hpssfile**
- **get "| localshell's input" : hpssfile**

  e.g tar pipes
- **put "| tar –cvf - ." : hpssfile**
- **get "| tar –xvf - " : hpssfile**
- *(but note htar is generally faster than these pipes)*

HSI AND FILTERS

- Some HSI commands generate ASCII output useful in filters
- Ls –l |grep 'string' or find . –print >localfile are SCHEMATIC examples.

BUT!!

HSI writes its verbiage to stderr by default

hsi "ls" |wc will print ZEROES for length.

Use –P option to divert verbiage to stdout

(but then you can't use pipes in file transfers)

hsi –P "ls" | wc will generate the expected counts

HSI FILE MANAGEMENT

- Management of deep directory trees was difficult with pftp. This encouraged fewer larger files.
- HSI will make directory searches and small file management easier for users.
- THIS HAS HPSS CONSEQUENCES!
- Nameserver speed is ~1/10 JFS speed and ½ GPFS speed. Ls of many files can take significant time (1 sec/30 files)
- Recursive ls and find can take a LONG time.
- ***Limiting file NUMBER is still necessary!!***

NIFTY NUGGETS

- Hsi supports a hook to disable staging. This may be useful when pulling large numbers of large files which might fill the cache and cause aborts.
- Syntax is get –S localfile : hpssfile
- Get –C purges from cache after get. Useful if the sum of gotten file sizes exceeds available cache. Behavior is analagous to dmget;use;dmput on crays

NIFTY NUGGETS

- Partial gets. Get –O hpssoffset:localoffset:length localfile : hsifile.
- This pulls hsifile starting at byte hpssoffset and begins writing to localfile at byte localoffset. Length can also be specified.
- If the byte offset on the local file is nonzero, a SPARSE FILE is created. Ls –l on such a file shows a longer length than is actually allocated. Only the bytes from localoffset are actually allocated.
- NCEP doesn't have much experience with these.
- Caution. Cp, dump, tar, backup, etc.. Make these DENSE!!
- Expect primary use will be to specify hpssoffset and length and leave localoffset alone
- These could have potential e.g. htar on GPFS.

NIFTY NUGGETS

- The ls –V command shows tape number, file offset on tape and other useful info (esp. for admins)

Keyword settings

- Some HSI settings are controlled with keywords.
- Most salient example is “set cosid”
- “set backup” supports a backup copy of overwritten files. Appears to be similar to single level version numbers on other O.S. (DEC VMS for example)
- Ten directories can be set dir0 defaults to home.
- *Have not prepared examples because of current BUG!! Set makes hsi segfault!!*

Some (temporary) caveats

- Wildcards are expanded by hsi. rm * can do a lot of damage and rm –rf * can do more.
- This is not a bug but a likely early user mistake

Some (temporary) caveats

- We have encountered a bug with keyword settings.
- Class of Service is specified with a keyword setting.
- This would be set cosid = “232” in hsi shell or script
- But changing any keyword setting with “set” causes segfaults as of 11/14
- HSI can only use size based defaults for file puts. Stdin puts (from pipes) default to “huge” *(I don’t know so make it big seems to be the choice)*
- We will get this addressed. Check ibmdocs and motd for updates.

Some (temporary) caveats

- Some find commands don't work as expected. A date bracket (find . –type f –mtime -5 –mtime +1) does not return files within the time bracket (two to four days old). We will get this addressed.
- Find –exec {} also doesn't work.
- Files written by pipe puts are sometimes unusable for a few minutes and then are fine. We're looking into this.
- ***There are likely others which we'll shake out.***

Persistent caveats

- HSI enables large numbers of files.
- HPSS nameserver service rate remains limited.
- Try to keep fewer larger files.

- HSI (and pftp) support a "stage" command which unmigrates files to disk cache.
- HSI wildcards make this MUCH more powerful. Stages of many many files can be generated with wildcards causing slow response for everyone else
- (but dmget * did same thing on crays).. Just be careful and considerate of other users.

SUMMARY

- HSI provides a more unix shell like interface to HPSS (but with some limitations)
- Pipes between HPSS and frost/snow commands are enabled.
- File management is MUCH easier but HPSS constrains practical file numbers.
- There are some teething pains with this release. Watch for documented and new bugs.

Wrapup

- This presentation supplements other documentation.
- Purpose was to provide a quick introduction for NCEP users.
- It is a work in progress. These slides will be placed on ibmdocs as a PDF and updated frequently. We will also place a slideshow on ibmdocs which will update less frequently.
- Check date on slide 1 or page 1.